

Санкт-Петербургский политехнический университет Петра Великого
Высшая школа электроники и микросистемной техники

Отчёт

Дисциплина: Цифровые устройства обработки сигналов

Тема: БИХ-фильтр с распараллеливанием

Студент группы 4941104/20701

П. П. Колбенков

Преподаватель

Доц.

М. С. Енученко

Сдано на проверку
« 27 » сентября 2023 г.

Принято
« _____ » _____ 2023 г.

Санкт-Петербург 2023

Задание

В данной работе согласно варианту задания необходимо создать математическую модель полосового БИХ-фильтра, работающего с данными с фиксированной точкой. В таблице 1 представлены исходные данные.

Таблица 1 – Исходные данные задания

Название блока	Разрядность входных/выходных отсчетов	Требования
Полосовой БИХ-фильтр с распараллеливанием х3	11	Ширина полосы 0.15, центр $f_s/4$, переходная полоса 0.05, подавление 65 дБ, неравномерность 0.25 дБ

Теоретические основы

Цифровой фильтр – устройство, работающее в реальном времени, принимающее на вход последовательность отсчётов и проводящее некоторую обработку данного сигнала. Вид обработки может быть различным, цифровой фильтр может выделять или подавлять определённые частоты во входном сигнале. Цифровой фильтр, как и аналоговый фильтр характеризуется своей передаточной и импульсной характеристикой. Структурно цифровые фильтры просты, основные блоки, использующиеся для построения цифровых фильтров – это сумматоры, умножители и блоки задержки, общая структурная схема цифрового фильтра приведена на рисунке 1.

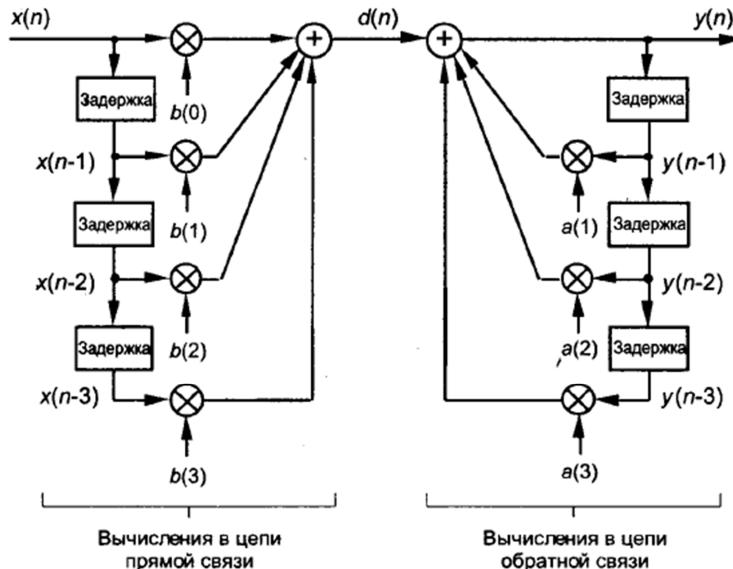


Рисунок 1 – Структурная схема цифрового фильтра [1]

Схема цифрового фильтра универсальна, характеристики фильтра, например его передаточная характеристика, определяется используемыми коэффициентами в умножителях. В самом простом случае, в случае КИХ-фильтра, коэффициенты для умножителей соответствуют дискретной импульсной характеристике фильтра. К сожалению, для формирования схемы БИХ-фильтра, то есть нахождения его коэффициентов, нет таких же простых методов, как для КИХ-фильтров, для создания БИХ-фильтров используются следующие методы: метод инвариантного преобразования импульсной характеристики, метод билинейного z-преобразования и оптимизационный метод.

Нахождение коэффициентов фильтра

Для нахождения коэффициентов фильтра воспользуемся встроенным в программную среду MATLAB набором инструментов fdatool. Данный набор инструментов позволяет по заданным требованиям к виду АЧХ или ФЧХ найти коэффициенты фильтров различных типов: Баттервorta, Чебышева, Эллиптический – чем мы и воспользуемся. Задав в окне инструмента требования к полосе пропускания, полосам подавления, величине неравномерности передаточной характеристики в полосе пропускания и величине подавления создадим фильтр. Как можно заметить, требования были заданы с небольшим избытком относительно требуемых по заданию, это сделано специально, для того чтобы в дальнейшем при процессе огрубления коэффициентов, в процессе перевода коэффициентов с двойной точностью к типу с фиксированной точкой и ограниченной разрядности, АЧХ фильтра всё ещё удовлетворяла требованиям. Рассмотрев доступные варианты типов фильтра, выбор был сделан в пользу эллиптического фильтра, так как данный тип фильтра реализовывал заданные требования при наименьших аппаратных затратах (наименьший порядок фильтра), окно программного инструмента fdatool приведено на рисунке 2.

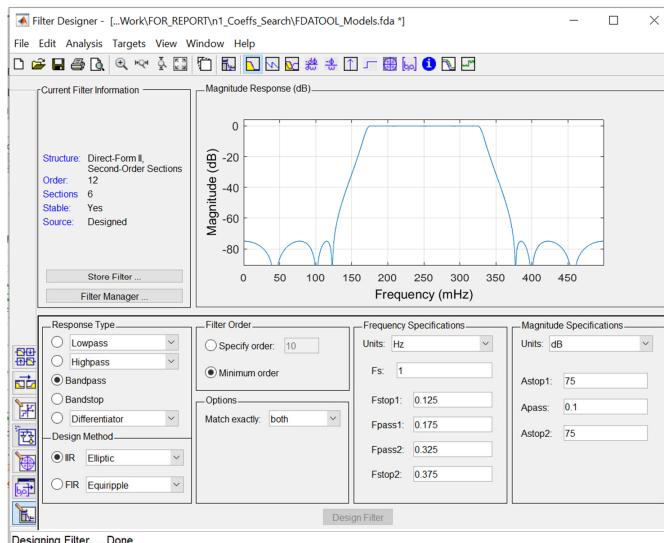


Рисунок 2 – Меню инструмента по созданию фильтров

В результате синтеза фильтра был получен фильтр двенадцатого порядка. При синтезе эллиптического фильтра fdatool использует метод билинейного z-преобразования. Для использования полученных коэффициентов требуется их экспортовать в MATLAB, что можно сделать в виде матрицы коэффициентов фильтра из секций второго порядка, которые затем можно преобразовать к виду, который будет использоваться в будущем – к набору a и b -коэффициентов для цепи обратной и прямой связи соответственно функцией sos2tf. Полученные коэффициенты можно использовать для построения передаточной характеристики фильтра, с помощью функции freqz.

Задание требует разработать модель фильтра, работающую с числами ограниченной размерности, в виде числа с фиксированной точкой. Для перевода полученных коэффициентов фильтра из двойной точности к числу с фиксированной точкой был написан matlab-скрипт, проверяющий соответствие АЧХ фильтра требованиям по его a и b -коэффициентам, а затем был написан скрипт в цикле, вызывающий предыдущий последовательно, уменьшая длину как всего числа, так и длину дробной части. Содержаний скриптов приведено в приложении А. Приведем результат расчёта коэффициентов и их огрубления в таблице 2.

Таблица 2 – Коэффициенты БИХ-фильтра

Номер коэффициента	a_i		b_i	
	Из fdatol	После огрубления	Из fdatol	После огрубления
1	-1	-1.000000000000	0.0045	0.00439453125
2	6.6613e-16	0.000000000000	-1.6006e-17	0.000000000000
3	-3.2912	-3.29150390625	-0.0106	-0.0107421875
4	1.8873e-15	0.000000000000	1.3005e-17	0.000000000000
5	-5.4561	-5.45605468750	0.0186	0.0185546875
6	1.5543e-15	0.000000000000	-1.2004e-17	0.000000000000
7	-5.4140	-5.41406250000	-0.0215	-0.0214843750
8	6.66133e-16	0.000000000000	-1.2004e-17	0.000000000000
9	-3.3545	-3.35449218750	0.0186	0.0185546875
10	0	0.000000000000	1.4005e-17	0.000000000000
11	-1.2184	-1.21826171875	-0.0106	-0.0107421875
12	-5.5511e-17	0.000000000000	-1.6006e-17	0.000000000000
13	-0.2031	-0.203125000000	0.0045	0.00439453125

В результате поиска было обнаружено что минимальная длина коэффициентов a , при которых выполняются требования к АЧХ составляет 15, а дробная длина 11, для коэффициентов b с учётом знаковости дробная длина составляет 11, а полная длина 12.

Приведём график АЧХ фильтра полученного из fdatool, и АЧХ после огрубления с наложенной маской требований (рисунки 3-4).

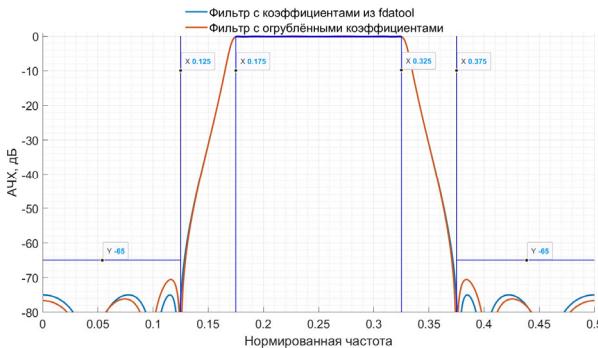


Рисунок 3 – Сравнение АЧХ фильтра до и после огрубления

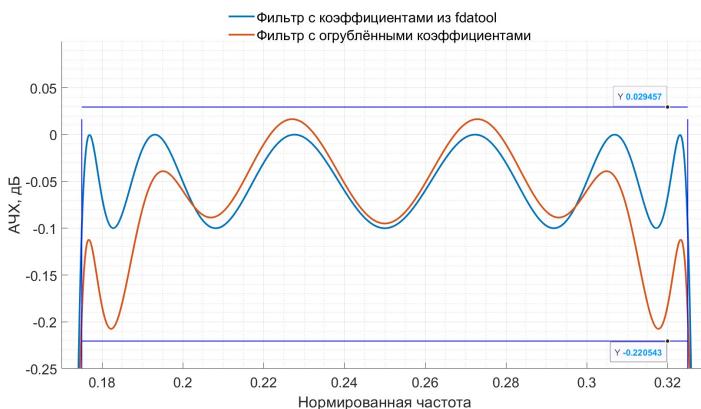


Рисунок 4 – Сравнение АЧХ фильтра до и после огрубления (полоса пропускания
увеличенено)

После огрубления коэффициентов неравномерность коэффициента передачи в полосе пропускания составила 0.224 дБ, а подавление составило 70.4 дБ. Можно заметить, что запас по подавлению оказался слишком велик, а значит можно синтезировать фильтр меньшего порядка изначально, но это не так, потому что, во-первых, коэффициенты фильтра подбирались утилитой Filter Designer, в ней было выставлена настройка минимального порядка, то есть до огрубления коэффициентов нельзя было создать фильтр с такими же характеристиками меньшего порядка, а во-вторых, стоит отметить, что большее ограничение при огрублении коэффициентов на минимальный порядок фильтра было у значения неравномерности в полосе пропускания, что также ограничило минимальный

порядок фильтра и его подавление, с учётом синтеза фильтра с запасом для дальнейшего ухудшения параметров огрублением коэффициентов.

Разработка эталонной модели

После того как мы нашли коэффициенты требуется разработать модель фильтра в Simulink. Для дальнейшего удобства распараллеливания отдельно рассмотрим КИХ-часть (не содержащую цепей обратной связи) и БИХ-часть фильтра (содержащую цепь обратной связи). Приведём на рисунках 5–7 структурную схему фильтра до распараллеливания.

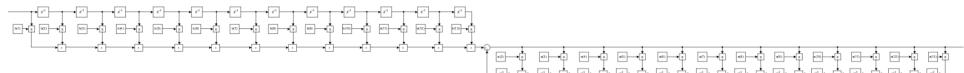


Рисунок 5 – Структурная схема фильтра

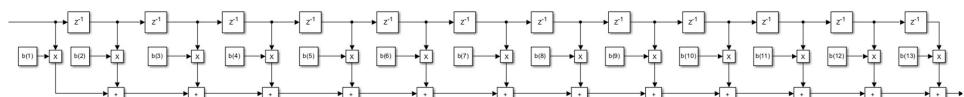


Рисунок 6 – Структурная схема фильтра (КИХ часть увеличено)

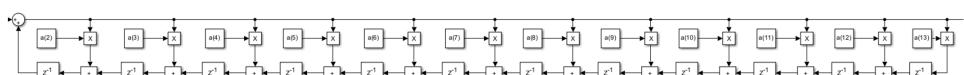


Рисунок 7 – Структурная схема фильтра (БИХ часть увеличено)

Для проверки правильности работы фильтра проведем моделирование и сравним полученные данные с результатом моделирования блока Discrete Filter встроенного в Simulink, позволяющего промоделировать фильтр по его коэффициентам. Результат моделирования приведён на рисунке 8.

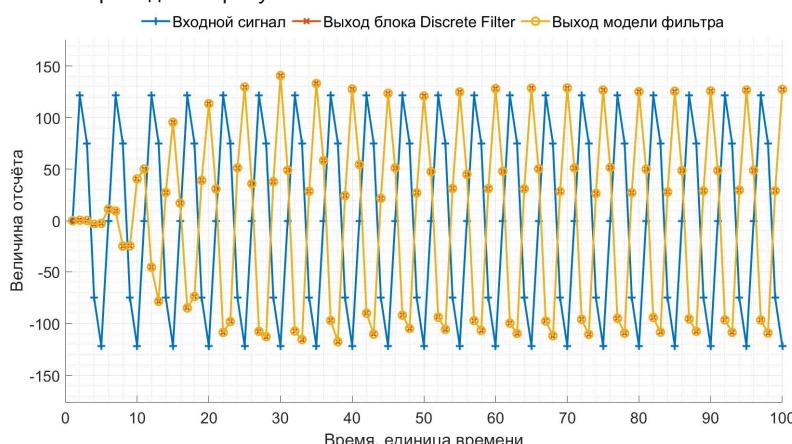


Рисунок 8 – Результат моделирования фильтра до распараллеливания

Результат моделирования фильтра совпадает с результатом моделирования блока Discrete Filter.

Распараллеливание фильтра

Начнём распараллеливание с КИХ-части фильтра, приведём её на рисунке 9.

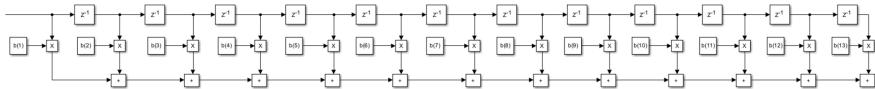


Рисунок 9 – КИХ-часть фильтра (первая стадия)

Для начала требуется три раза повторить линейку задержек, и подключить выводы задержек к умножителям через каждый третий блок задержки, из-за чего получатся линейки из трёх последовательно соединённых блоков задержки без подключений между ними. Данные линейки из трёх блоков задержки Z^{-1} преобразуются в блоки Z^{-3} , что показано на рисунке 10.

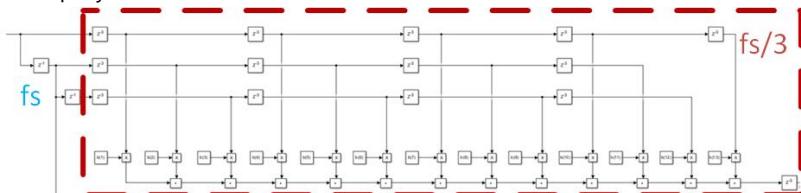


Рисунок 10 – КИХ-часть фильтра (вторая стадия)

Последний шаг к распараллеливанию – повторение полученной структуры три раза (рисунок 11). Для получения сигнала с выхода данной части фильтра с частотой дискретизации равной входной требуется провести мультиплексирование, выстроить выходные отсчёты с частотой дискретизации в три раза меньшей чем на входе последовательно. На данном этапе мультиплексирование не проводится, так как последующая БИХ-часть фильтра также распараллелена, поэтому принимает на вход три потока данных с частотой дискретизации в три раза меньшей входной.

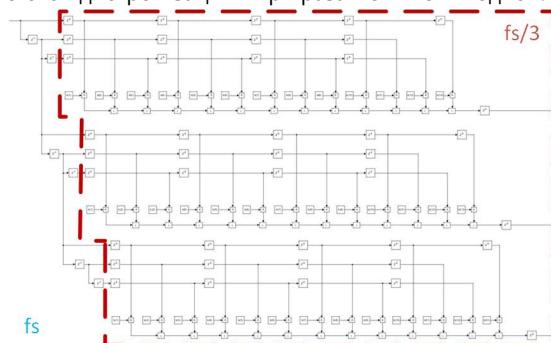


Рисунок 8 – КИХ-часть фильтра (третья стадия)

Рассмотрим далее процесс распараллеливания БИХ-части. Во многом данная операция аналогична распараллеливанию КИХ-части фильтра, но из-за наличия петель обратной связи нет возможности просто заменить все задержки Z^{-1} на Z^{-3} напрямую, поэтому требуется внести в схему перекрёстные соединения между каждым из трёх «экземпляров» фильтра.

На рисунке 12 приведена схема БИХ-части фильтра до распараллеливания, на рисунке 13 приведен этап распараллеливания с повторением линий задержки и объединением задержек между собой.

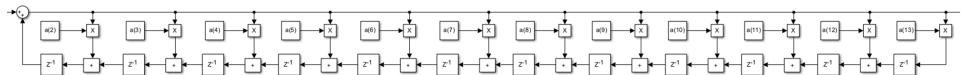


Рисунок 12 – БИХ-часть фильтра (первая стадия)

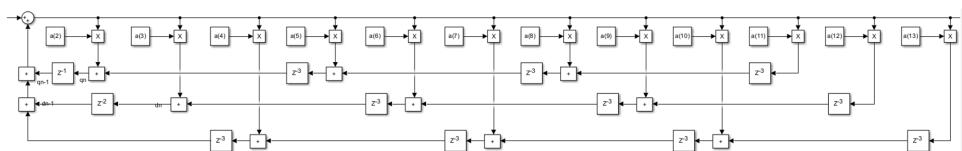


Рисунок 13 – БИХ-часть фильтра (вторая стадия)

Как можно видеть из рисунка 13, при параллелизации БИХ-структурой появляются задержки Z^{-1} и Z^{-2} , которые нельзя заменить в рамках одного «экземпляра» фильтра. Назовём выход сумматора после умножения на $a(2) q_n$, выход последующей задержки q_{n-1} , соответственно, а выход следующего сумматора d_n и выход последующей задержки d_{n-2} соответственно. Повторим получившуюся структуру три раза со смещением на тakt каждого экземпляра посредством добавления задержки (рисунок 14).

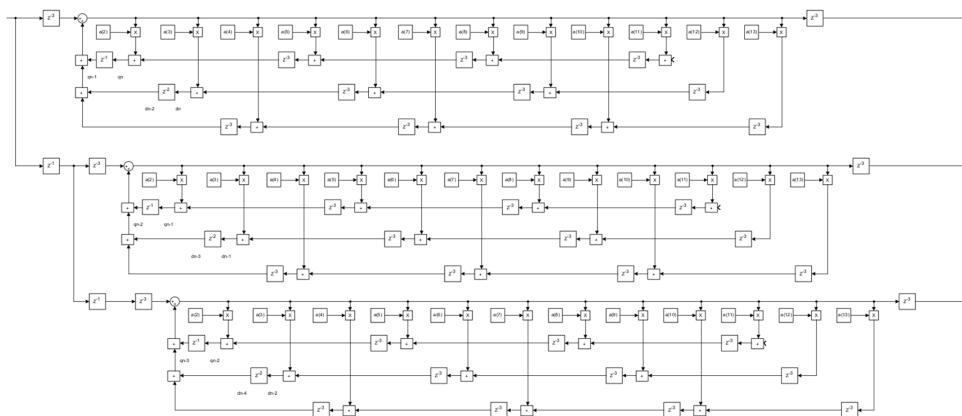


Рисунок 14 – БИХ-часть фильтра (третья стадия)

Заметим, что названные ранее выводы сумматоров в других экземплярах фильтров будут сдвинуты на так, так мы получим ещё $q_{n-2}, q_{n-3}, d_{n-3}, d_{n-4}$. А значит соединим их между

собой попарно, где разница составляет три такта (d_{n-1} и d_{n-4}), можно использовать задержки Z^{-3} , что и продемонстрировано на рисунке 15. Так как мы избавились от блоков задержки не равных Z^{-3} то можно понизить частоту, что и продемонстрировано на рисунке 15.

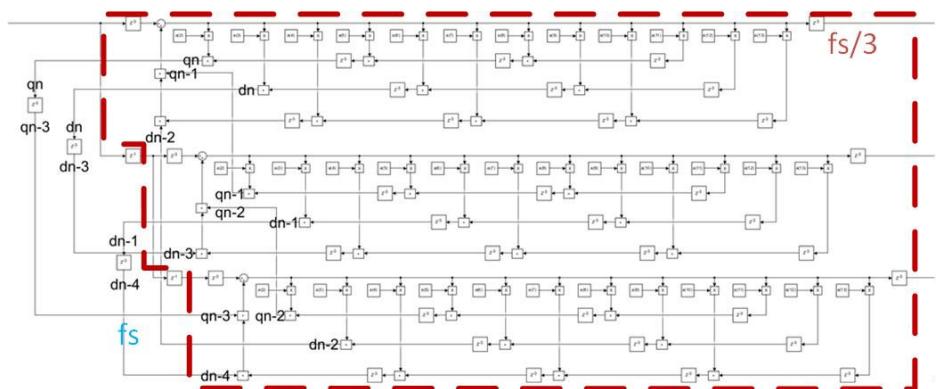


Рисунок 15 – БИХ-часть фильтра (четвёртая стадия)

Как было сказано ранее при распараллеливании части фильтра с конечной импульсной характеристикой, преобразование потока данных с выхода распараллеленной части с понижением частоты дискретизации требует мультиплексирования, которое выполняется с помощью трехходового мультиплексора с подключённым к нему счётчиком с коэффициентом пересчёта равным трем и частотой счёта равной частоте дискретизации входного потока данных для фильтра. Данная конструкция, повышающая частоту дискретизации, приведена на рисунке 16.

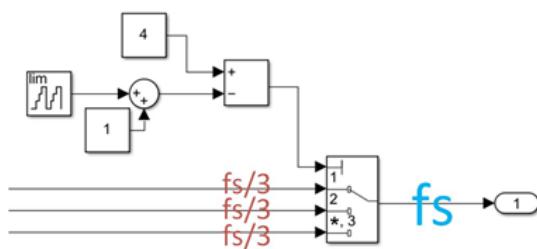


Рисунок 16 – Мультиплексор на выходе

Упрощение модели и создание библиотеки элементов

Целью данной работы является создание модели фильтра, работающего с ограниченной точностью в каждом блоке. Так как в модели фильтра используется много повторяющихся блоков, из-за параллелизации, была создана библиотека элементов, чтобы

можно было один раз изменив характеристики блока в библиотеке, автоматически поменять характеристики всех блоков в модели. Изображение библиотеки приведено на рисунке 17.

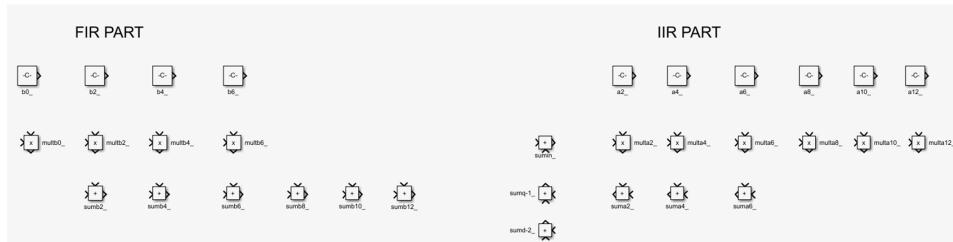


Рисунок 17 – Библиотека элементов

Можно заметить, что в библиотеку, также были внесены коэффициенты фильтра, в виде констант для удобства изменения, также можно заметить, что присутствуют только четные коэффициенты фильтра, и блоки умножения на данные коэффициенты, это сделано так, потому что после огрубления до конечной разрядности коэффициентов получилось, что коэффициенты с нечётными номерами равны нулю, что также упростило модель, что будет показано далее.

Если рассмотреть КИХ-часть можно заметить, что количество коэффициентов и блоков умножения в два раза меньше чем в БИХ-части при том же порядке, сделано это было так, потому что в КИХ-части коэффициенты симметричны относительно среднего, то есть попарно равны, получается что и блоки умножений будут использованы одинаковые, ведь блок умножения работает с входным отсчетом и коэффициентом, а значит размерности парных блоков умножения будут равны.

Если рассмотреть БИХ часть, то можно заметить блоки сумматоров $sumq-1$ и $sumd-2$, это те самые блоки сумматоров перекрёстной связи «экземпляров» фильтров в параллельном фильтре. На рисунке 15 можно увидеть, что перекрёстных сумматоров связи шесть штук, но если рассмотреть то, к каким блокам присоединены данные сумматоры, можно заметить, что у каждого « d »-сумматора входные значения – выводы сумматора sum_{ab} и сумматора sum_{a2} , а у « q »-сумматора входные значения – выводы сумматора sum_{a4} и сумматора $sumd-2$. Отсюда можно сделать вывод, что из шести сумматоров связи только два являются уникальными, остальные являются копиями этих двух, что позволяет внести блоки сумматоров $sumq-1$ и $sumd-2$ в библиотеку для дальнейшего множественного использования.

Также после объединения частей фильтров было принято решение мультиплексировать выводы с параллельных ветвей фильтра только в самом конце, а между КИХ и БИХ-частью соединить ветви напрямую, что, во-первых, убирает потребность в дополнительном мультиплексоре между частями фильтра, а, во-вторых, убирает потребность в дополнительных блоках задержки между ветвями после мультиплексирования. По одному блоку задержки между КИХ и БИХ частью было вставлено для разрыва большого критического пути. Также во избежание проблем при переходе из

более медленного временного домена в более быстрый, был использован блок Zero-Order-Hold на выходе параллельных ветвей БИХ-части фильтра.

На выходе используется переключаемый ключ с счётчиком с коэффициентом пересчёта 3, что выполняет работу мультиплексирования отсчётов для восстановления сигнала на более высокой частоте дискретизации чем той, на которой работают блоки фильтров.

Покажем изменённую модель фильтра после упрощений на рисунках 18-20.

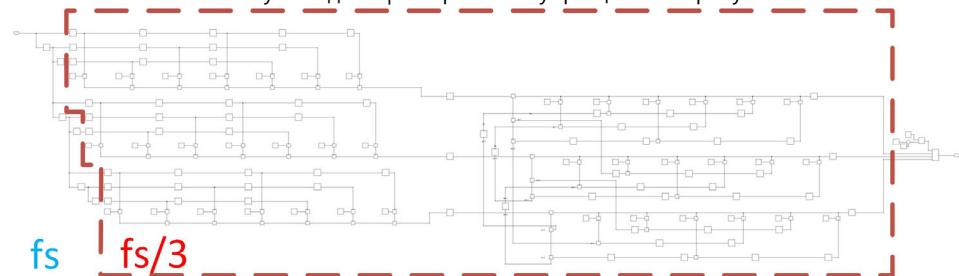


Рисунок 18 – Полная модель фильтра

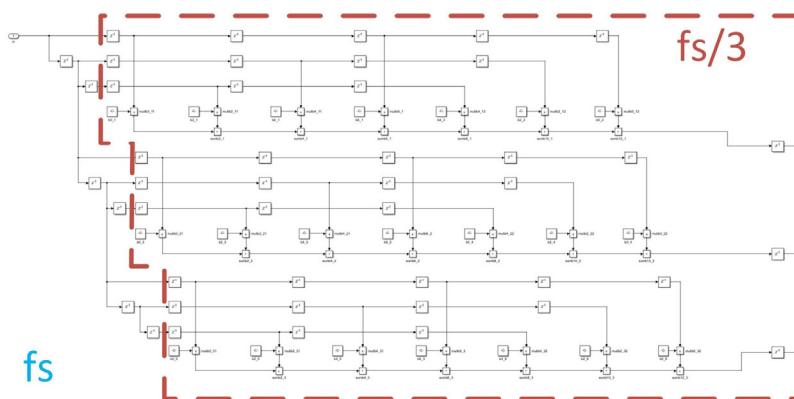


Рисунок 19 – КИХ-часть после упрощений

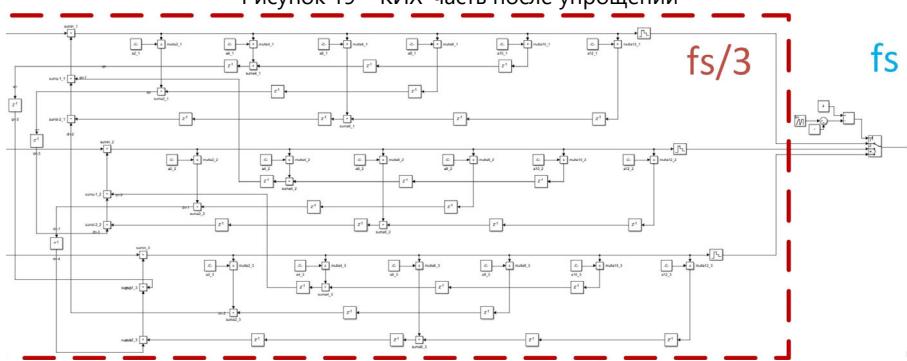


Рисунок 20 – БИХ-часть после упрощений

Сразу же найдем критический путь комбинационной логики, ограничивающий максимальное быстродействие схемы фильтра. Критический путь начинается и заканчивается на блоке задержки – то есть на блоке памяти. Рассмотрим обе части фильтра на рисунках 21 и 22.

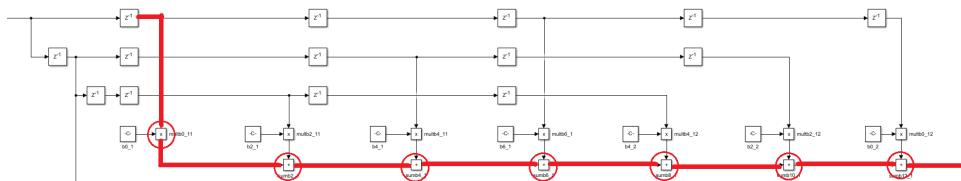


Рисунок 21 – Критический путь в КИХ-части

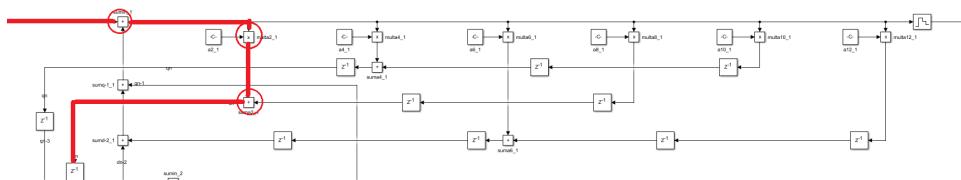


Рисунок 22 – Критический путь в БИХ-части

Как видно на приведённых рисунках, в КИХ части критический путь длиннее, и составляет один умножитель и шесть сумматоров.

Критический путь можно сократить конвейеризацией, чтобы увеличить быстродействие схемы, но увеличивая задержку между входными и выходными данными блока, но это не было задачей данной работы.

Преобразование модели к фиксированной точке

Для нахождения оптимальных разрядностей блоков умножения и сложения было создано тестовое окружение в Simulink, приведём его на рисунке 23.

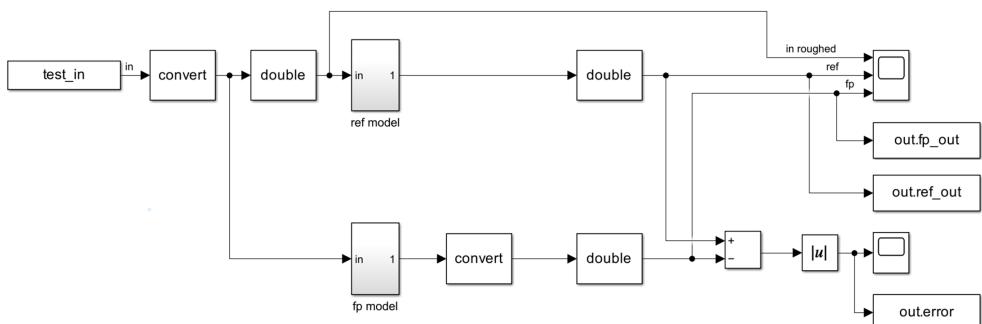


Рисунок 23 – Тестовое окружение в Simulink

Данное тестовое окружение ожидает на вход из MATLAB сигнал *test_in* в двойной точности, затем блоком *convert* сигнал преобразуется к фиксированной точке с заданной заданием разрядностью и некоторой длиной дробной части, далее сигнал поступает на *fp model* – модель фильтра в фиксированной точке, и на *ref model* – эталонную модель, после обратного преобразования сигнала в двойную точность. Эталонная модель представляет из себя копию модели с фиксированной точкой, только все блоки математический операций в ней имеют выходное значение *double*. После прохождения блоков фильтров сигнал снова преобразуется в двойную точность с ограничением до требуемой по заданию разрядности (только выход *fp model*), затем сигналы выводятся в MATLAB и на экран с помощью блока *Scope*, также выходные сигналы фильтров вычитаются для нахождения разницы между моделями, то есть ошибки, данный сигнал также выводится в MATLAB для дальнейшего анализа.

Для автоматизации поиска оптимальных разрядностей блоков был написан скрипт, который итерационно уменьшает разрядность выбранного блока умножения или сложения, при этом проводя каждый раз моделирование тестового окружения и находит максимальное отклонение в выходном сигнале фильтра фиксированной точности от эталонной модели, и когда разница составляет более 0.5 МЗР для любого из тестовых входных сигналов останавливает поиск. Тестовые сигналы в данном случае являлись синусоидальные сигналы с тремя различными частотами, равными началу полосы пропускания, фильтра, середине полосы пропускания фильтра, и частоте с максимальным коэффициентом передачи в полосе пропускания. Ограниченностю набора объясняется тем, что моделирование занимает некоторое время, а в процессе перебора приходится проводить моделирование множество раз, что приводит к большому времени поиска, отсюда и стремление сократить набор тестовых сигналов, но сократить его ещё сильнее не представляется возможным, из-за неравномерности коэффициента передачи фильтра в полосе пропускания, что приводит к тому, что подобрав разрядность блоков фильтра на одной частоте, на другой может случиться переполнение.

Таким образом, начиная с эталонной модели, выбирался один блок, с помощью скрипта для данного блока была найдена предельная разрядность, далее выставив для исследованного блока найденную разрядность переходила работа к следующему. Так были найдены сперва разрядности умножителей, затем разрядности сумматоров каждой из частей фильтра.

Учитывалось, что при детальном рассмотрении АЧХ фильтра, в полосе пропускания можно было обнаружить отрезки на котором фильтр оказывал небольшое усиление, что приводило к тому, что на выходе фильтра длина целой части увеличивалась на единицу по сравнению со входом.

Содержание скрипта приведено в приложении Б.

Приведём получившиеся разрядности блоков в таблице 3

Таблица 3 – Выбранные разрядности блоков сумматоров и умножителей

Часть фильтра	Наименование блока	Количество знаковых бит	Полная разрядность блока, бит	Длина дробной части, бит
КИХ	multb0	1	22	21
	multb2	1	21	20
	multb4	1	21	20
	multb6	1	20	19
	sumb2	1	22	21
	sumb4	1	22	21
	sumb6	1	22	21
	sumb8	1	22	21
	sumb10	1	22	21
	sumb12	1	22	21
БИХ	multa2	1	39	36
	multa4	1	44	40
	multa6	1	29	25
	multa8	1	36	33
	multa10	1	26	24
	multa12	1	27	26
	suma2	1	33	29
	suma4	1	42	38
	suma6	1	28	24
	sumin	1	36	34
	sumq-1	1	39	35
	sumd-2	1	34	30

В таблице приведена максимальная длина дробной части при максимальной длине дробной части входного сигнала.

Поиск производился генерацией сигнала с максимальной амплитудой для выбранной разрядности, то есть, при условии, что по заданию разрядность одиннадцать бит, максимальная амплитуда – два в степени десять (с учётом знака) минус один с умножением на два в отрицательной степени раной длине дробной части, в процессе поиска она принималась также максимальной равной десяти. После каждого нахождения разрядности проверялась работа фильтра во всём диапазоне полосы пропускания во избежание дальнейших проблем из-за неравномерности коэффициента передачи.

Проверка работоспособности блока на различных частотах и амплитудах входного сигнала

Для проверки работоспособности блока выберем следующий набор амплитуд, допустим разрядность входной шины равна четырём битам, тогда с учётом знаковости

максимальная амплитуда в двоичном виде равна 111_2 на порядок (в два раза) меньшая амплитуда равна 11_2 и так далее. Поступим аналогичным образом для шины размером 11 бит, протестируем блок на одиннадцати различных амплитудах, отличающихся на порядок. Частоты выберем равномерно распределенными по всем частотному диапазону от 0 до 0.5 в нормированной частоте. Проведем моделирование блока на данных частотах и амплитудах входных сигналов, приведём результат в таблице 4. Длина дробной части была принята равно длине целой части – 5 бит.

Таблица 4 – Величина ошибки при различных параметрах входного сигнала

	Нормированная частота								
Амплитуда	0.0000	0.0625	0.1250	0.1875	0.2500	0.3125	0.3750	0.4375	0.5000
1111111111_2	0,000000	0,480232	0,484375	0,495030	0,491301	0,495030	0,484375	0,480232	0,000000
111111111_2	0,000000	0,478619	0,292190	0,498004	0,496289	0,498004	0,292190	0,478619	0,000000
11111111_2	0,000000	0,239309	0,145809	0,499997	0,412228	0,499997	0,145809	0,239309	0,000000
1111111_2	0,000000	0,487994	0,072619	0,499078	0,452366	0,499078	0,072619	0,487994	0,000000
111111_2	0,000000	0,326991	0,036023	0,498968	0,360668	0,498968	0,036023	0,326991	0,000000
11111_2	0,000000	0,063234	0,017726	0,359314	0,354515	0,359314	0,017726	0,063234	0,000000
1111_2	0,000000	0,273188	0,008577	0,495287	0,171539	0,495287	0,008577	0,273188	0,000000
111_2	0,000000	0,458803	0,004003	0,486879	0,080052	0,486879	0,004003	0,458803	0,000000
11_2	0,000000	0,216493	0,001715	0,469821	0,034308	0,469821	0,001715	0,216493	0,000000
1_2	0,000000	0,369407	0,000639	0,498987	0,011436	0,498987	0,000639	0,369407	0,000000

Максимальная ошибка при всех параметрах составляет 0,499997 в единицах МЗР.

Для каждого значения из таблицы был построен график с результатом временного анализа, так как общее количество рисунков велико, они приведены в прилагаемой директории «\Демонстрация работоспособности фильтра», там можно ознакомиться с результатами, в названии файлов отражены характеристики подаваемого входного сигнала, на графике указана максимальная ошибка относительно эталонной модели. Приведём три примера: подавление сигнала в левой полосе подавления (рисунок 24), сигнал в полосе пропускания (рисунок 25) и подавление сигнала в правой полосе подавления (рисунок 26).

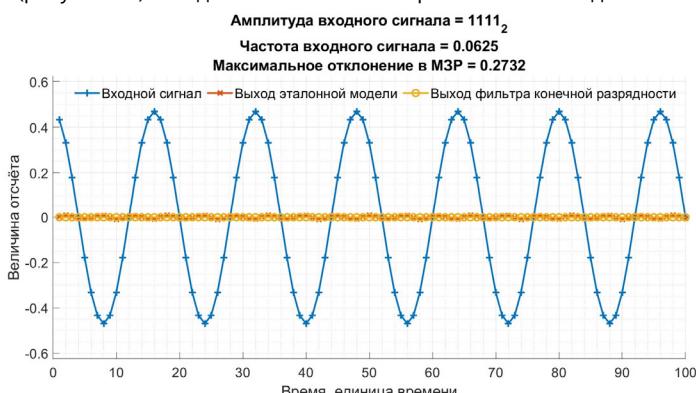


Рисунок 24 – Временной анализ работы фильтра (подавление сигнала в левой полосе подавления)

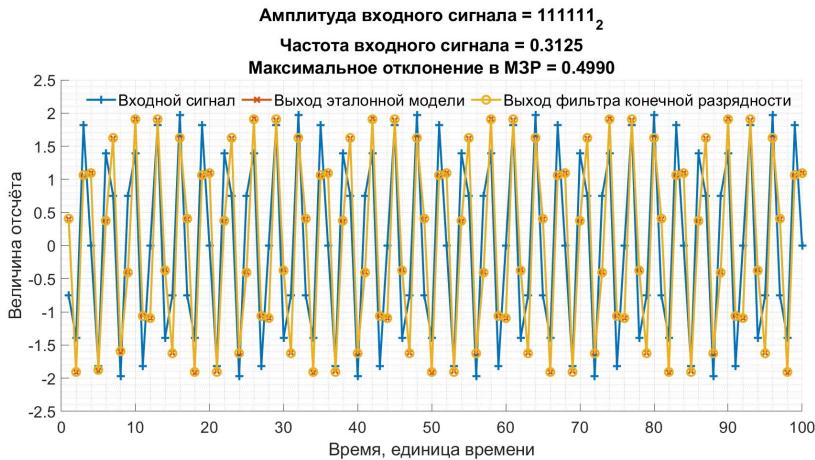


Рисунок 25 – Временной анализ работы фильтра (сигнал в полосе пропускания)

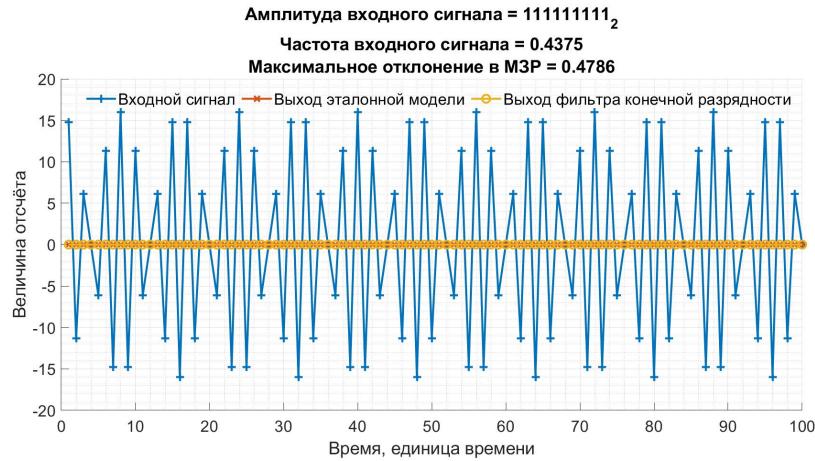


Рисунок 26 – Временной анализ работы фильтра (подавление сигнала в правой полосе подавления)

Проверка отсутствия избыточности в разрядностях математических блоков фильтра

Проверим нет ли избыточности в выборе разрядностей для блоков математических операций. Попробуем уменьшить на единицу разрядность нескольких случайных блоков, запустим моделирование на наборе частот в полосе пропускания, проверим как изменится максимальная ошибка относительно эталонной модели. В данном случае на вход фильтра подаётся синусоидальный сигнал определённой частоты с максимальной амплитудой, находится максимальное отклонение относительно от эталонной модели, эксперимент

повторяется для другой частоты, максимальные отклонения в зависимости от частоты сигнала приведены на графиках.

Приведём график ошибки от частоты до изменений на рисунке 27. А также пример результата временного анализа фильтра при подаче частоты с максимальным отклонением от эталонной модели (рисунок 28).

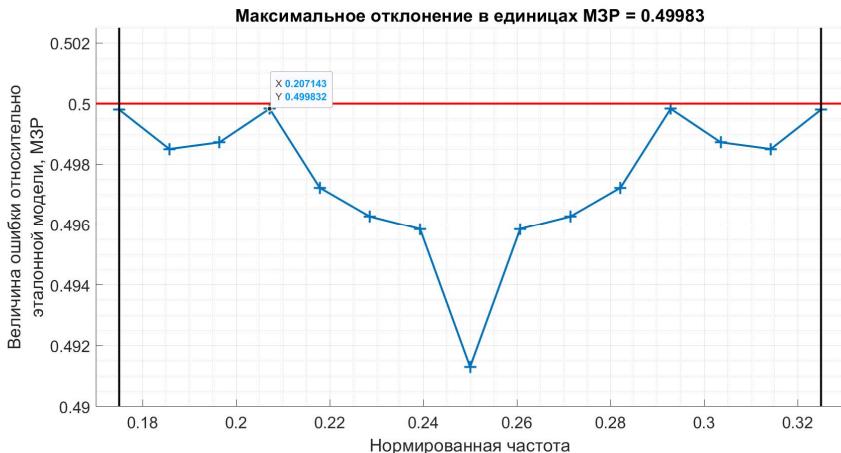


Рисунок 27 – Зависимость ошибки от частоты до изменений

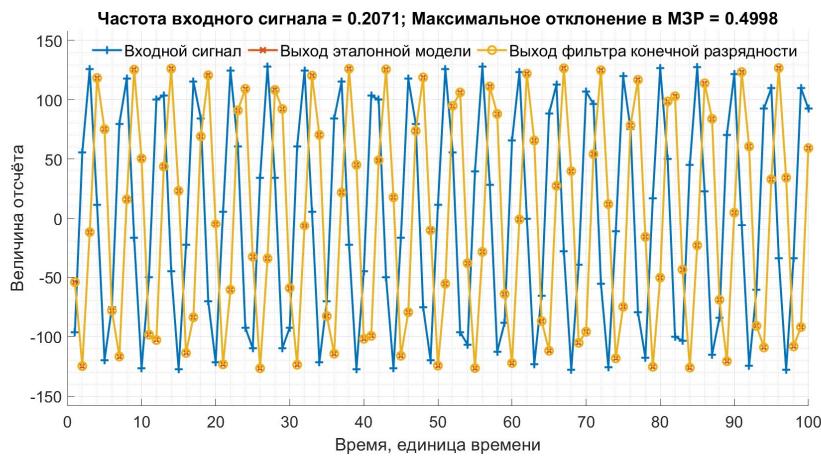


Рисунок 28 – Результат временного анализа до изменений

Начнём проверку минимальности размерностей блоков математических операций с КИХ-части фильтра. Если рассматривать блоки математических операций КИХ-части, то можно заметить, что у них у всех длина дробной части равна разрядности блока с учётом знака для случая, когда длина дробной части входных отсчётов равна разрядности входных отсчётов с учётом знака. Таким образом, чтобы не прибегать к ненормальной длине дробной части, можно уменьшить только длину дробной части. Уменьшим длину дробной

части умножителя *multb2* КИХ-части на единицу, результат приведём на рисунке 29, результат временного анализа фильтра при подаче частоты с максимальным отклонением от эталонной модели на рисунке 30.

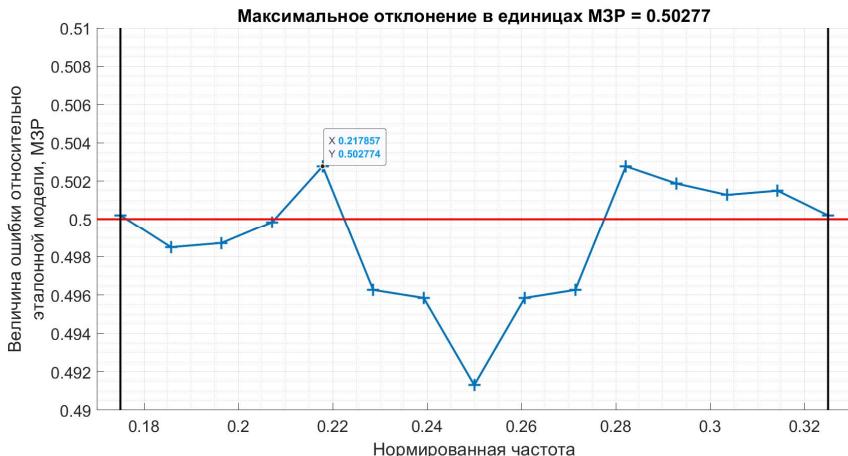


Рисунок 29 – Зависимость ошибки от частоты после уменьшения длины дробной части *multb2*

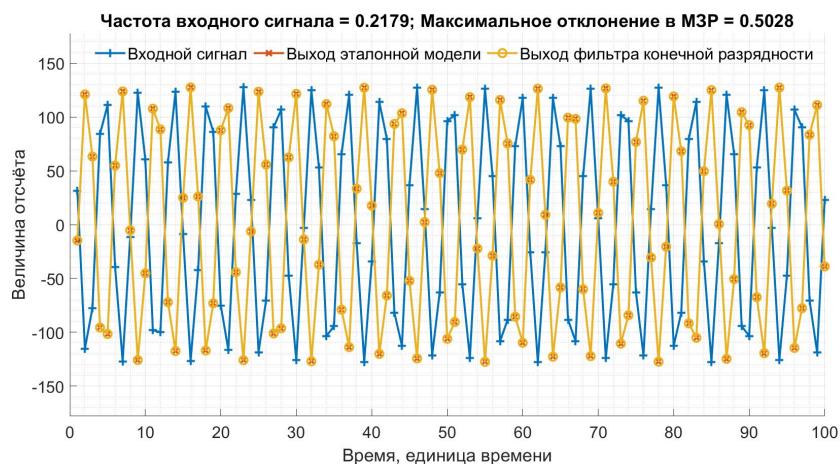


Рисунок 30 – Результат временного анализа после уменьшения длины дробной части *multb2*

Уменьшим длину дробной части сумматора *sumb8* КИХ-части на единицу, результат приведём на рисунке 31, результат временного анализа фильтра при подаче частоты с максимальным отклонением от эталонной модели на рисунке 32.

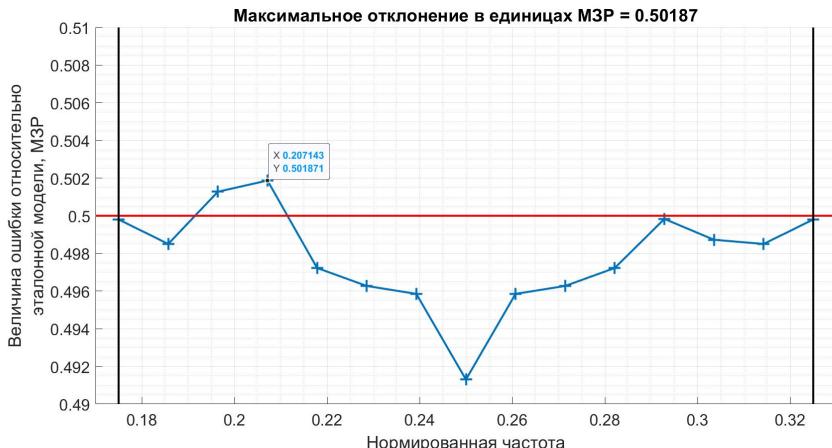


Рисунок 31 – Зависимость ошибки от частоты после уменьшения длины дробной части $sumb8$

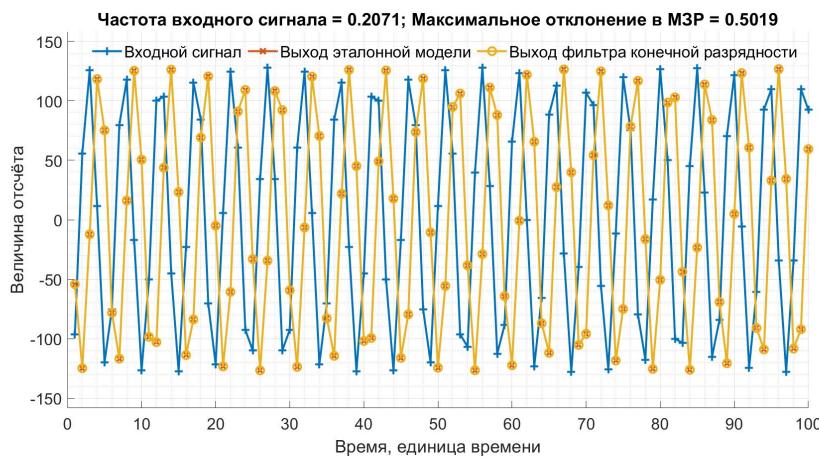


Рисунок 32 – Результат временного анализа после уменьшения длины дробной части $sumb8$

Уменьшив на единицу разрядность умножителя $multa10$ БИХ-части, результат приведём на рисунке 33, результат временного анализа фильтра при подаче частоты с максимальным отклонением от эталонной модели на рисунке 34.

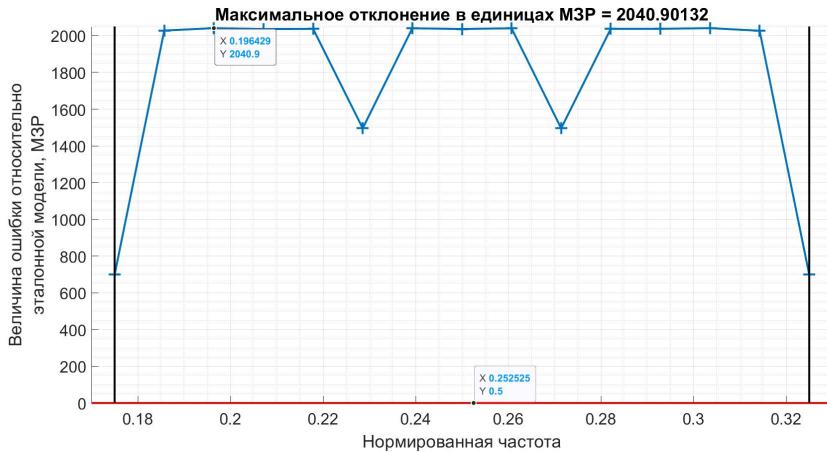


Рисунок 33 – Зависимость ошибки от частоты после уменьшения длины целой части *multa10*

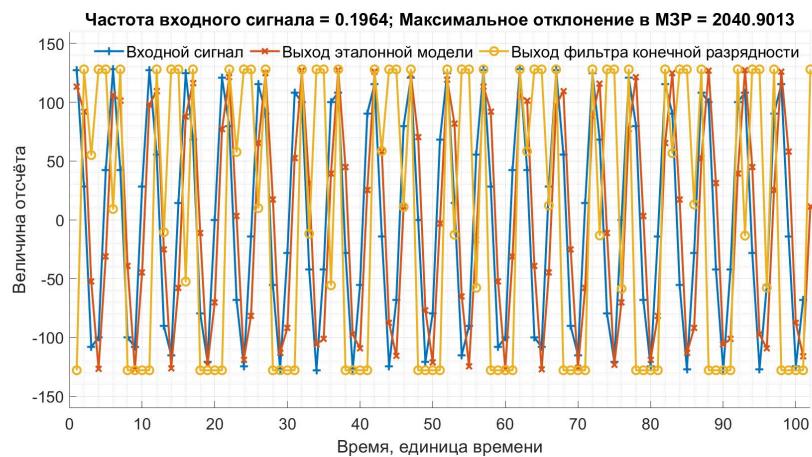


Рисунок 34 – Результат временного анализа после уменьшения длины дробной части *multa10*

Уменьшив длину дробной части умножителя *multa10* БИХ-части на единицу, результат приведём на рисунке 35, результат временного анализа фильтра при подаче частоты с максимальным отклонением от эталонной модели на рисунке 36.

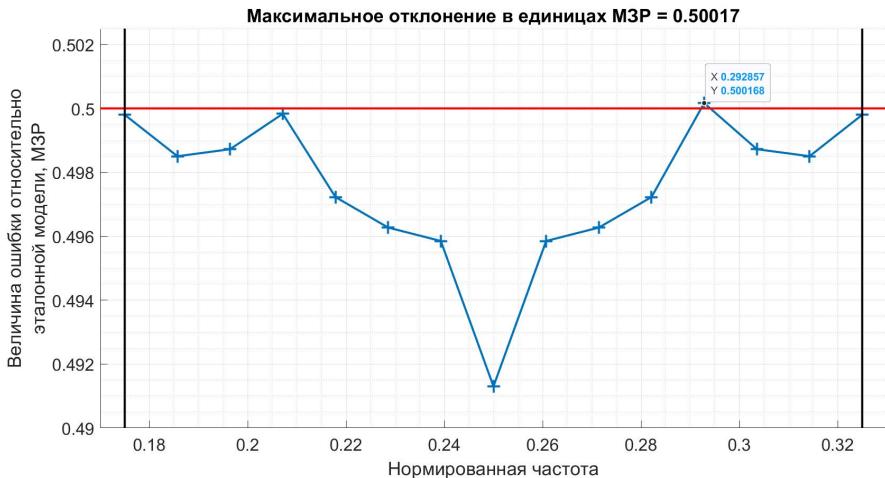


Рисунок 35 – Зависимость ошибки от частоты после уменьшения длины дробной части $multa10$

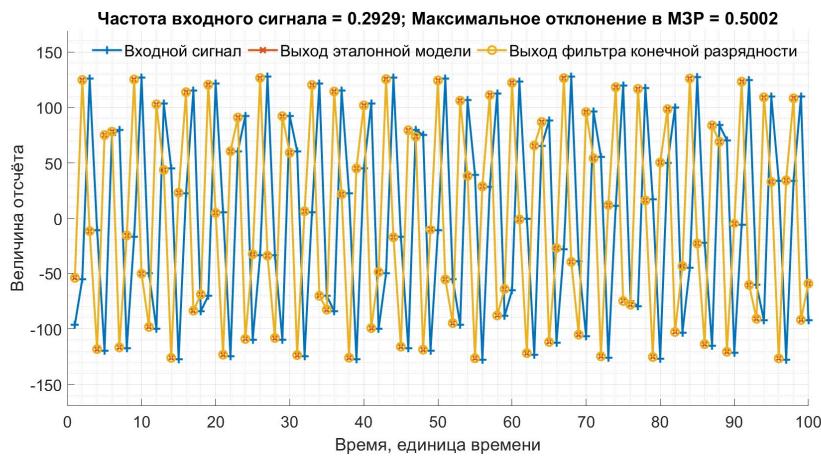


Рисунок 36 – Результат временного анализа после уменьшения длины дробной части $multa10$

Уменьшим на единицу разрядность умножителя $suma2$ БИХ-части, результат приведём на рисунке 37, результат временного анализа фильтра при подаче частоты с максимальным отклонением от эталонной модели на рисунке 38.

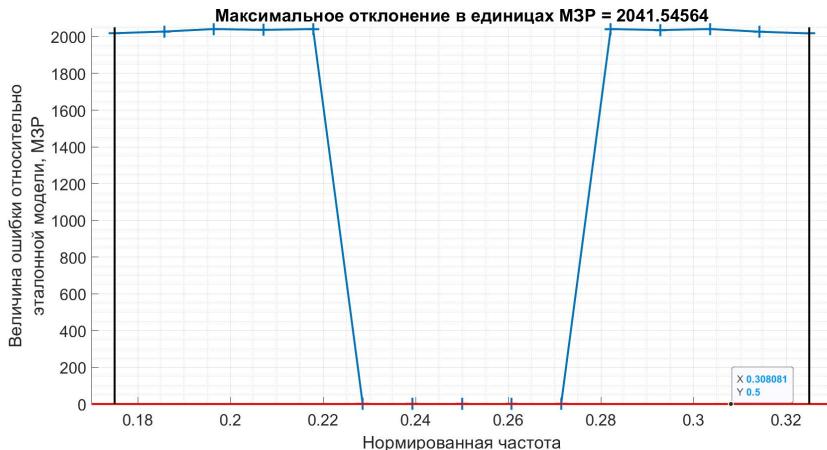


Рисунок 37 – Зависимость ошибки от частоты после уменьшения длины целой части *sumat2*

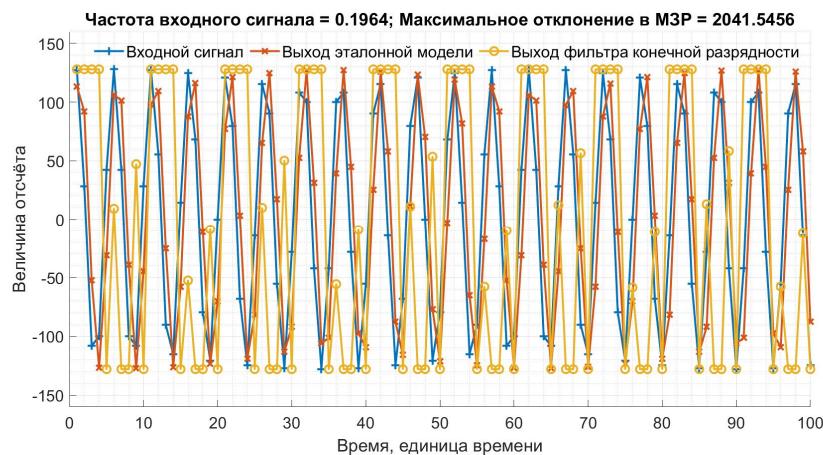


Рисунок 38 – Результат временного анализа после уменьшения длины дробной части *multa10*

Уменьшим длину дробной части умножителя *sumat2* БИХ-части на единицу, результат приведём на рисунке 39, результат временного анализа фильтра при подаче частоты с максимальным отклонением от эталонной модели на рисунке 40.



Рисунок 39 – Зависимость ошибки от частоты после уменьшения длины дробной части $suma2$

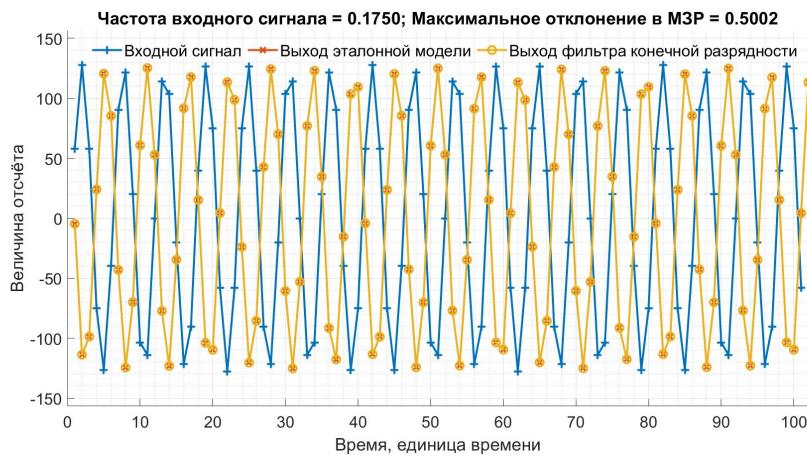


Рисунок 40 – Результат временного анализа после уменьшения длины дробной части $multa10$

Нахождение чувствительности фильтра

Если рассмотреть таблицу номер четыре, можно заметить, что даже при самой маленькой амплитуде сигнала величина ошибки составляет меньше половины МЗР, проверим что это фильтр действительно способен работать на такой амплитуде, подробнее рассмотрев случай такой амплитуды. Промоделируем работу фильтра на наборе частот в полосе пропускания фильтра, аналогично тому, как было сделано в прошлом пункте, только амплитуда сигнала принимается не максимальной, а минимальной, равной одному биту, приведём результат на рисунках 41-42.

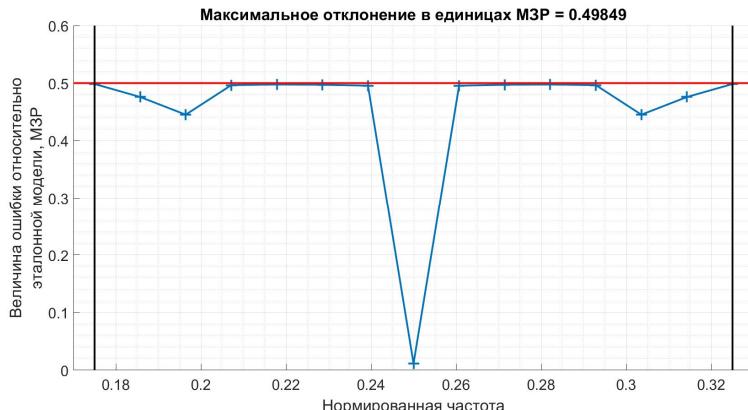


Рисунок 41 – Зависимость ошибки от частоты при минимальной амплитуде сигнала

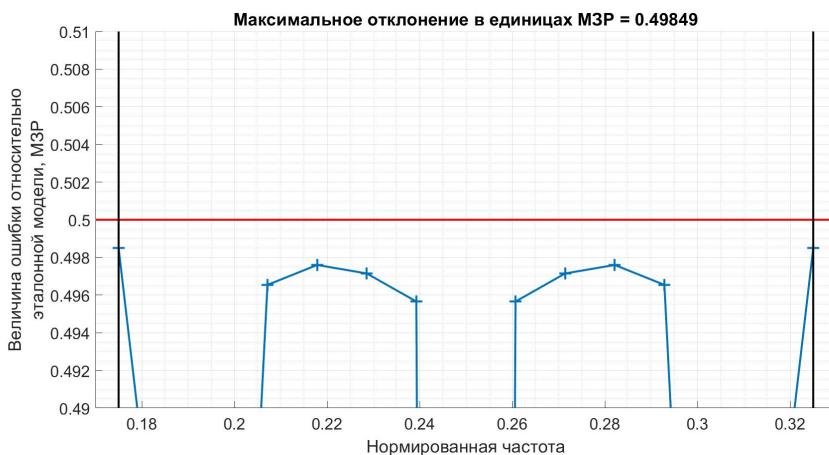


Рисунок 42 – Зависимость ошибки от частоты при минимальной амплитуде сигнала
(увеличено)

По приведённым графикам можно утверждать, что фильтр способен обрабатывать сигнал с минимальной амплитудой. Приведём несколько графиков – результатов временного анализа работы фильтра при подаче на вход сигнала минимальной амплитуды различной частоты на графиках 43-45. С большим количеством графиков для других частот, но с минимальной амплитудой можно ознакомиться в прилагаемой директории, о которой было написано ранее. Для простоты восприятия, длина дробной части входных сигналов на приведённых графиках равна нулю.

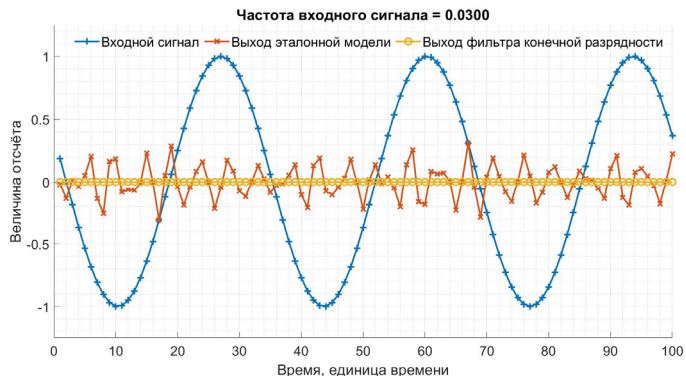


Рисунок 43 – Временной анализ работы фильтра (подавление сигнала в левой полосе подавления)



Рисунок 44 – Временной анализ работы фильтра (сигнал в полосе пропускания)

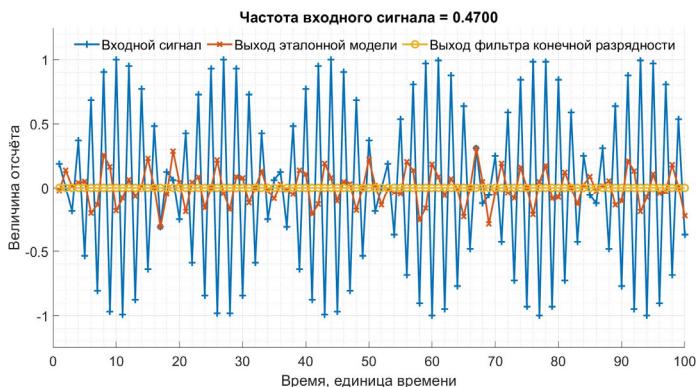


Рисунок 45 – Временной анализ работы фильтра (подавление сигнала в правой полосе подавления)

Моделирование АЧХ фильтра

Используя инструмента Curve Fitting, встроенного в MATLAB, можно определить амплитуду выходного сигнала фильтра, что позволяет найти соотношение между входной и выходной амплитудой сигнала для фильтра, так можно провести моделирование АЧХ фильтра. Проведём моделирование работы фильтра с набором сигналов различной частоты, сравним выходную амплитуду сигнала со входной, результат приведём на рисунках 46-47.

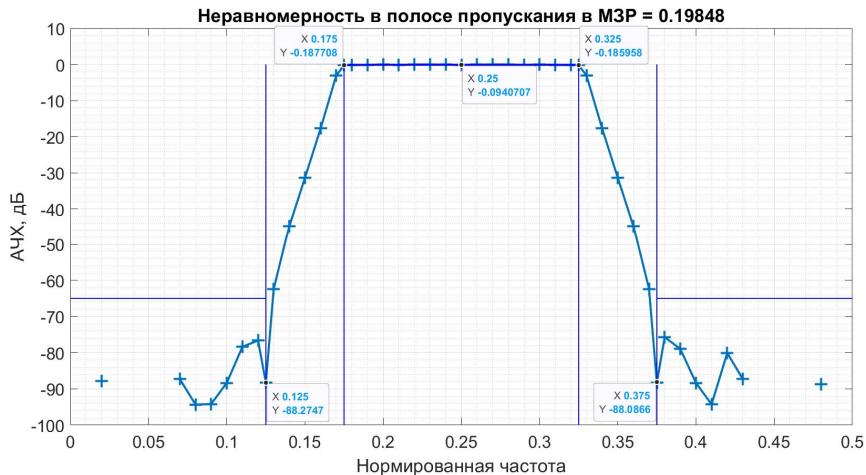


Рисунок 46 – АЧХ фильтра

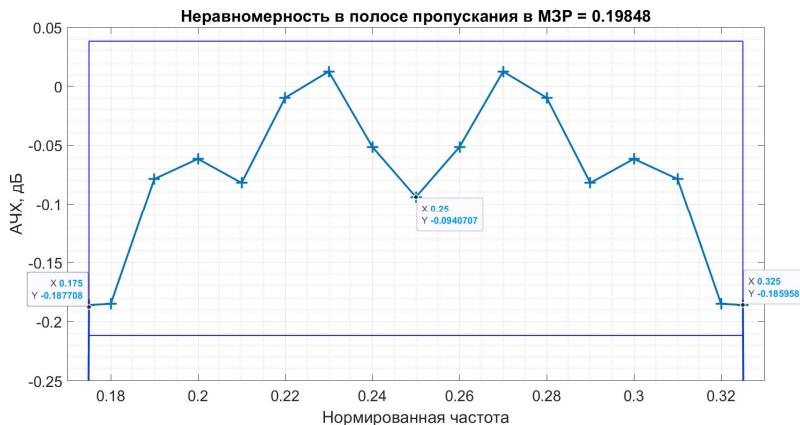


Рисунок 47 – АЧХ фильтра (полоса пропускания)

По АЧХ, приведённой на рисунках 46-47, укладывающейся в маску требований можно судить о правильности реализации фильтра.

Содержание скрипта для получения АЧХ фильтра приведено в приложение В.

Литература

1. Lyons R.G. Understanding Digital Signal Processing / R.G. Lyons. – 3rd ed. – Boston: Prentice Hall, 2011. – pp. 787.

ПРИЛОЖЕНИЕ А

А.1 Проверка соответствия требованиям по АЧХ фильтра

```
function [bool,rip,l_att,r_att] = IS_MATCHING(a_fi,b_fi)
%Function checks if filter with roughed coefficients still
% meets desired requirements
[h,w] = freqz(b_fi,a_fi,'whole',4000);
freq_n = w/(2*pi); % Вектор нормализованных частот
freq_resp =mag2db(abs(h)); % АЧХ фильтра по коэффициентам

pb_w = 0.15; % passband width
pb_c = 0.25; % passband center
trb_w = 0.05; % transitionband width
r = 0.25; % ripple db
att = 65; % attenuation db

pb_b = pb_c-pb_w/2; % passband begining
pb_e = pb_c+pb_w/2; % passband end

sb_le = pb_b-trb_w; % stopband left end 0.0 --sb-- l_end -pb- r_end --sb-- 0.5
sb_re = pb_e+trb_w; % stopband right end

%----- ripple check

i_bw_b = find(freq_n > pb_b,1) - 1; %index of passband beginning
i_bw_e = find(freq_n > pb_e,1) - 1; %index of passband end

pb_M = freq_resp(i_bw_b:i_bw_e);

pb_max_M = max(pb_M);
pb_min_M = min(pb_M);
rip = abs(pb_max_M-pb_min_M);

%----- sb att check

i_sb_le = find(freq_n > sb_le,1); %index of left stopband end
i_sb_re = find(freq_n > sb_re,1)-1; %index of right stopband end
i_fs2 = find(freq_n > 0.5,1) - 1; % index of fs = 0.5

sb_l_M = freq_resp(1:i_sb_le); % M for Magnitude (freq response)
sb_r_M = freq_resp(i_sb_re:i_fs2);

l_att = abs(max(sb_l_M));
r_att = abs(max(sb_r_M));

if (rip<=r) & (l_att>=att) & (r_att>=att) % Если найденное подавление в полосах подавления и
%неравномерность в полосе пропускания соответствуют заданию, функция возвращает true
    bool = true;
else
    bool = false;
end;
end
```

А.2 Итеративный поиск минимальной размерности коэффициентов

```
%Исходя из значений коэффициентов a и b были заданы следующие начальные предположения о word
% и fractional length коэффициентов a, 24, 20 b, 21, 20
a_intl = 4;
%a_wl_0 = 24;
```

```

a_f1_0 = 21;
%b_wl_0 = 21;
b_f1_0 = 20;

sum_res = 24+20;

for i_a = 0:15 % Цикл последовательного уменьшения длины дробной части коэффициентов а
    for i_b = 0:15 % Цикл последовательного уменьшения длины дробной части коэффициентов б

        a_f1 = a_f1_0 - i_a;
        a_wl = a_intl + a_f1;
        b_f1 = b_f1_0 - i_b;
        b_wl = b_f1+1;

        a_fi = fi(a, true, a_wl, a_f1); % Перевод коэффициентов в представление с
        фиксированной точкой, тем самым огрубляя коэффициенты
        a_fi = double(a_fi); % Перевод коэффициентов из представления с фиксированной точкой
        в плавающую с сохранением огрублённости
        b_fi = fi(b, true, b_wl, b_f1); % Перевод коэффициентов в представление с
        фиксированной точкой, тем самым огрубляя коэффициенты
        b_fi = double(a_fi); % Перевод коэффициентов из представления с фиксированной точкой
        в плавающую с сохранением огрублённости
        [bool,ripple,l_atten,r_atten] = IS_MATCHING(a_fi,b_fi); % Проверка соответствия
        требованиям после огрубления
        if bool
            sum_res = a_wl+b_wl;
            fprintf('a_wl %d b_wl %d\n', a_wl, b_wl);
            fprintf('ripple = %.3f l_attenuation = %.1f r_attenuation =
%.1f\n\n', ripple,l_atten,r_atten);
        end;
    end;
end

```

ПРИЛОЖЕНИЕ Б

Б.1 Скрипт автоматического поиска оптимальной разрядности математического блока

```
% Для начала надо в параметрах исследуемого блока написать следующее
% выражение в графе Output fixdt(1,wlx,flx-10+FL)
clear;
close all force;

freq_vec = [0.175; 0.225; 0.25]; % Набор тестовых частот
WL_vec = zeros(size(freq_vec));
FL_vec = zeros(size(freq_vec));
ERR_vec = zeros(size(freq_vec));
%-----Test signal generation
WL = 11; % Установка размерности входных отсчётов по заданию
FL = 10; % Установка максимальной длины дробной части входных отсчётов
A_0 = (2^(WL-1)); % Установка максимально возможной амплитуды сигнала для заданной
разности отсчётов с учётом знака
A_0 = A_0 * 2^(-FL); % Учёт дробной части

N = 250; % Кол-во отсчётов
Ts = 1; % Период дискретизации
T = Ts * N; % Время моделирования
t = 0:Ts:T*(N-1); % Вектор времени
fs = 1/Ts; % Частота дискретизации

for i = 1:length(freq_vec) % Цикл по всем тестовым частотам
    f_0 = freq_vec(i)*fs;
    FLout = FL;
    if (0.221<f_0)&&(f_0<0.233)|| (0.267<f_0)&&(f_0<0.279) % Учёт отрезков положительного
коэффициента передачи фильтра (требуется увеличить длину целой части выходных отсчётов,
уменьшив длину дробной части)
        FLout = FL-1;
    end
    lsb_val = 2^(-FLout);
    x_0 = A_0 * sin(2*pi*f_0.*t); % Тестовый сигнал на данной итерации
    test_in = [t', x_0'];

    sign_bit = 1;
    intx = 5; % Начальное предположение о длине целой части выходного значения блока
    flx = 40; % Начальное предположение о длине дробной части выходного значения блока

    %-----FRAC LEN SEARCH
    SIM_and_ERR_PRINT; % Запуск скрипта на моделирование и нахождения максимального
отклонения от эталонной модели
    while err_lsb < 0.5 % Пока не превышен порог по ошибке уменьшается длина дробной части и
 заново моделируется фильтр
        flx = flx - 1;
        SIM_and_ERR_PRINT;
    end
    flx = flx + 1;
    %-----INT LEN SEARCH
    SIM_and_ERR_PRINT; % Запуск скрипта на моделирование и нахождения максимального
отклонения от эталонной модели
    while err_lsb < 0.5 % Пока не превышен порог по ошибке уменьшается длина целой части и
 заново моделируется фильтр
        intx = intx-1;
        SIM_and_ERR_PRINT;
        if wlx <= flx
            break
        end
    end
```

```

    end
    intx = intx + 1;
%-----CHECK
SIM_and_ERR_PRINT; % Проверка и фиксация найденной размерности и длины дробной части
ERR_vec(i) = err_lsb;
WL_vec(i) = wlx;
FL_vec(i) = flx;
end

%-----RESULT
flx = max(FL_vec);
if max(WL_vec)<=flx
    wlx = flx+1;
else
    wlx = max(WL_vec);
end

Multiple_CHECK_sc; % Проверка найденной размерности и длины дробной части для более широкого
 набора частот

fprintf('WL = %d    FL = %d\n', wlx, flx);

freq_vec = [0.175; 0.225; 0.25];
colNames = {'Freq', 'WL', 'FL', 'Err in lsb'};
RES = [freq_vec WL_vec FL_vec ERR_vec];
Restable = array2table(RES,'VariableNames',colNames)

fprintf('max err at all freqs = %f\n', max(err_vec));

```

Б.2 Вызываемый скрипт для моделирования (SIM_and_ERR_PRINT)

```

wlx = sign_bit+intx+flx;
simout = sim("REF_VS_FL.slx");
error = simout.error.Data;
err_lsb = max(error(150:end))/lsb_val;
fprintf('WL = %d    FL = %d    ERR = %f\n', wlx, flx, err_lsb);

```

Б.3 Вызываемый скрипт для проверки (Multiple_CHECK_sc)

```

WL = 11;
FL = 3;
INT = WL - FL;
A_0 = (2^(WL-1)-1); %Max amp
A_0 = A_0 * 2^(-FL);

freq_vec = zeros(16,1);
pb_center = 0.25;
pb_w = 0.15;
freq_vec(1) = pb_center - pb_w/2;
freq_delta = pb_w/(length(freq_vec)-1);
for i = 2:length(freq_vec)
    freq_vec(i) = freq_vec(i-1)+freq_delta;
end

N = 250;
Ts = 1;
T = Ts * N;
t = 0:Ts:Ts*(N-1);
fs = 1/Ts;

err_vec = zeros(size(freq_vec));
for i = 1:length(freq_vec)
    f_0 = freq_vec(i)*fs;

```

```

x_0 = A_0 * sin(2*pi*f_0.*t);
test_in = [t', x_0'];
FLout = FL;
if (0.221<f_0)&&(f_0<0.233)|| (0.267<f_0)&&(f_0<0.279)
    FLout = FL-1;
end
lsb_val = 2^(-FLout);
simout = sim("REF_VS_FI.slx");
error = simout.error.Data;
m_err = max(error(150:end));
err_vec(i) = m_err/lsb_val;
end
plot(freq_vec,err_vec)

```

ПРИЛОЖЕНИЕ В

В.1 Скрипт для моделирования АЧХ фильтра

```
clear;
close all force;

WL = 11; % Установка размерности входных отсчётов по заданию
FL = 3; % Установка арбитрной длины дробной части входных отсчётов
A_0 = (2^(WL-1)); % Установка максимально возможной амплитуды сигнала для заданной
размерности отсчётов с учётом знака
A_0 = A_0 * 2^(-FL);

freq_vec = linspace(0,0.5,51); % Вектор тестовых частот

N = 250; % Кол-во отсчётов
Ts = 1; % Период дискретизации
T = Ts * N; % Время моделирования
t = 0:Ts:T*(N-1); % Вектор времени
fs = 1/Ts; % Частота дискретизации

out_amp = zeros(size(freq_vec));
for i = 1:length(freq_vec)
    f_0 = freq_vec(i)*fs;
    x_0 = A_0 * sin(2*pi*f_0.*t); % Тестовый сигнал на данной итерации
    test_in = [t', x_0'];
    FOut = FL;
    if (0.221<f_0)&&(f_0<0.233)|| (0.267<f_0)&&(f_0<0.279) % Учёт отрезков положительного
коэффициента передачи фильтра (требуется увеличить длину целой части выходных отсчётов,
уменьшив длину дробной части)
        FOut = FL-1;
    end
    lsb_val = 2^(-FOut);
    w=warning('off','all');
    simout = sim("REF_VS_FI.slx"); % Запуск моделирования тестового окружения в Simulink
    warning(w);
    fp_out = simout.fp_out.Data;
    fp_out = fp_out(150:end-1);
    x = t(150:end)';
    y = fp_out;
    f = fit(x,y,'sin1'); % Нахождения амплитуды и частоты выходной сигнала фильтра для
нахождения отношения выходной амплитуды к входной
    out_amp(i) = f.a1;
end

freq_resp = mag2db(out_amp/A_0);

plot(freq_vec,freq_resp, '-+', 'MarkerSize', 12, 'LineWidth', 2)
xlabel('Нормированная частота');
ylabel('АЧХ, дБ');
grid on;
grid minor;

ripple = mask_plot(freq_vec, freq_resp);
formatSpec = '%.5f';
str = num2str(ripple,formatSpec);
str = "Неравномерность в полосе пропускания в МЗР = "+str;
title(str);
```